International Journal of Electronics and Communication Engineering (IJECE)
ISSN 2278-9901
Vol. 2, Issue 3, July 2013, 63-76
© IASET

International Academy of Science, Engineering and Technology
Connecting Researchers; Nurturing Innovations

PRIORITIZED FAULT REPORTING USING WIRELESS SENSOR NETWORKS IN INDUSTRIAL ENVIRONMENT

SHRIPAD V DESHPANDE[1] & P. R. DEVALE[2]

[1]Research Student, Department of Information Technology, Bharati Vidyapeeth University College of Engineering, Pune, Maharashtra, India

[2]Professor, Department of Information Technology, Bharati Vidyapeeth University College of Engineering Pune, Maharashtra, India

ABSTRACT

Wireless Sensor Networks (WSNs) are now established as one of the most cost effective and efficient mechanism for collecting data on the industrial shop floor. The sensors, apart from just collecting data, can also be used to detect faults occurring in the process being monitored by them. Traditionally, such faults are accumulated in the central node and processed collectively at one place. Instead of this, a distributed approach for fault processing is proposed. The faults can be temporarily stored in the sensor node and only selected faults can be sent on priority basis to the central node. The algorithm for filtering of the fault alarms is presented. The algorithm is tested in a simulated environment on RT-Linux platform.

KEYWORDS: Wireless Sensor Network, IWSN, Alarm Filtering, Industrial Process Control

INTRODUCTION

An Industrial Process is a dynamic entity. For proper and safe operation of the process, it must be closely monitored and any parameter going out of expected threshold value must be brought under control. Since last decade or so, Wireless Sensor networks have been emerging as a cost-effective and efficient technology to gather the real time process data. Equipped with a set of transducers, a microcontroller and a transceiver, a wireless sensor monitors the process unattended, creates self-configured ad-hoc, multi-hop network with neighboring sensor nodes and transfers the scanned data to respective higher nodes for further processing (Figure 1).

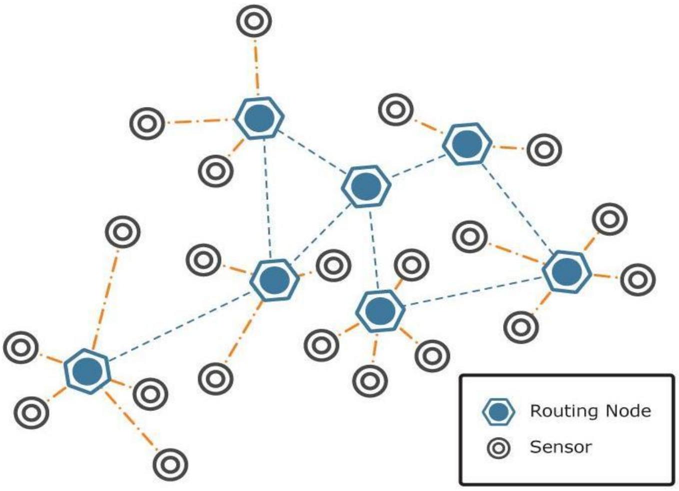

Figure 1: Basic Structure of Wireless Sensor Network

The widespread acceptance of Wireless Sensor Networks has taken a boost after the delivery of the IEEE 802.15.4 standard [1, 2] for the physical and MAC layers and the ZigBee standard for network and application layers. When deployed in large quantities in the field, these sensors ubiquitously acquire data and the system as a whole works like eyes and ears deeply penetrated into the complexity of the process. The alarm function is normally implemented in a central computer which after gathering data from all sensors, compares them with set threshold values and generates alarms. Many times in a complex industrial process where there is an interdependence of various processes on one another, it is common to get a flood of alarms for the same root cause. To filer out redundant alarms, alarm-filtering is used by co-relating them and removing the superfluous ones. Usually such an alarm filtering is done at the central node where all data is collected. Instead of doing the filtering at the central node, if it is done at the sensor itself or at least at the node which is one-level up - data gathering node, it will give many advantages – it will reduce the volume of data that travels to the central node, thus reducing the network traffic on the resource crunched sensor nodes; it will make it possible to diagnose the root cause faster and also it will reduce processing load on the central node.

Alarm filtering has been a field of active research for many years and various algorithms are proposed by researchers for this. Use of expert systems is also widespread in designing effective alarm processing in Industrial environment.

In this paper, we present an algorithm for alarm filtering at the sensor level and sink level in an industrial WSN. The rest of this paper is organized as follows – Section II briefly describes research work done so far in this field. Section III summarizes WSN structure in general and its role in Industrial Environment. The concept of Alarm filtering is explained in section IV and our algorithm is explained in section V. Section VI explains the simulated environment used for this research and section VII presents results of the work. Section VIII concludes the paper.

LITERATURE SURVEY

This work combines ideas from many diverse domains – Wireless Sensor Networks, Industrial Process Control, Adhoc and mesh networks, Alarm Filtering, Real Time Systems, Simulation (of WSNs) etc. Hence relevant literature from all these fields has been surveyed.

In [1], authors have focused on efficient approximate data collection with pre-specified error bounds in wireless sensor networks. The key idea of their data collection approach, ADC (Approximate Data Collection) is to divide a sensor network into clusters, and discover local data correlations on each cluster head. It then performs approximate data collection on the sink node according to model parameters uploaded by cluster heads.

In [2], authors propose a dynamic en-route filtering scheme that addresses both false report injection and DoS attacks in wireless sensor networks. The authors design the Hill Climbing key dissemination approach which they claim that ensures the nodes closer to data sources have stronger filtering capacity.

In [3], an energy-aware scheduling scheme for wireless sensor networks is proposed by the authors. The scheme they propose, is a periodic on–off scheduling scheme in which sensor nodes use only local information to make the scheduling decisions. The proposed scheme is checked in terms of detection delay, target hit rate, and energy consumption per successful target detection.

In [4], a comprehensive review of existing data compression approaches in wireless sensor networks is done. Suitable sets of criteria are identified to compare existing techniques as well as to make sure what practical data compression in wireless sensor networks should be. Later, the details of each classified compression category are described. And Finally, their performance, limitations, open issues and suitable applications are analyzed and compared based on the criteria of practical data compression in wireless sensor networks.

In [5], the authors discuss wireless sensor networks in industrial automation, focusing especially on performance issues, both in the design phase and during actual operation. This is a chapter from a book which is a compiled collection of WSN related documents.

In [6], the authors propose a proactive alarm reduction method whereby based on the contents of the past operating effects, alarm reduction is carried out during the next transient. They designed and implemented the proactive alarm reduction system and constructed the environment for the human factors validation test.

In [7], the authors suggest a set of tools which support operator discretion and initiative for alarm filtering. These tools are viewed as implemented on top of those properly detailed alarm displays and interlocks which reflect the more formal plant operating policies like alarm trending, alarm activity summary, causal alarm pattern analysis etc.

In [8], the article presents a new method for filtering out the un-important faults, by means of the defined dynamic fault tree (DFT). The proposed methodology includes the dependencies between fault events in the models. It tries to solve two problems: they relate to the filtering of false alarms, and then reduction of the size of the ambiguity of fault isolation related to the occurrence of a failure.

The authors claim that, in response to the need for diagnosis, and for the need for localization and filtering of the faults, it is necessary to introduce new dynamic gates, thus making it possible to translate new dependencies, relationships. The authors claim that the proposed DFT model can be modularized and each module translated into a High Level Petri Net (HLPN).

In [9], the authors develop a secure and distributed reprogramming protocol named SDRP, for tasking the WSN. The protocol uses identity-based cryptography to secure the reprogramming and to reduce the communication and storage requirements of each node.

In[10], the authors investigate differences of the behavior of WSN in real deployment compared to the results produced using simulator, which ignores the lower layers effects such as packet collision and overhearing. An open source tool ns2 is used which provides a complete protocol stack. Researchers explain the rationale of the variance of results produced by real deployment and that of simulators.

In [11], the authors propose to implement hybrid operating systems based on two-level hardware interrupts. This is of relevance because we have simulated WSN nodes in RT-Linux to get as Real Time response as possible. In [11], researchers propose to implement hybrid operating systems with two-level hardware interrupts by combining the real-time kernel and the timesharing OS (Operating System) kernel. They implement a hybrid system called RTLinux-THIN (Real-Time LINUX with Two-level Hardware INterrupts) on the ARM architecture by combining ARM Linux kernel 2.6.9 and uCOS-II.

In [12], the authors propose a scheme for en-route filtering of data in WSN, referred to as Grouping-enhanced Resilient Probabilistic En-route Filtering (GRPEF). Here, an efficient distributed algorithm is proposed to group nodes for deriving location-aware keys used to overcome the threshold problem and remove the dependence on the sink immobility and routing protocols. Compared to the existing schemes, this technique is claimed to significantly improve the effectiveness of the en-route filtering particularly in mobile sinks.

In [13], the researchers propose a novel distributed data structure called distributed data cube (DDC). This significantly helps in reducing network traffic by aggregating data in the form of special aggregate values (prefix sum, prefix average, prefix max, and prefix min) in distributed sensor nodes.

SENSOR NETWORKS

Structure of WSN

Equipped with a set of transducers, a microcontroller and a transceiver, a wireless sensor node monitors the process unattended, creates self-configured ad-hoc, multi-hop network with neighboring sensor nodes and transfers the scanned data to respective higher nodes for further processing. The widespread acceptance of Wireless Sensor Networks has taken a boost after the introduction of the IEEE 802.15.4 standard which specifies the physical and MAC layers and the ZigBee standard for network and application layers.

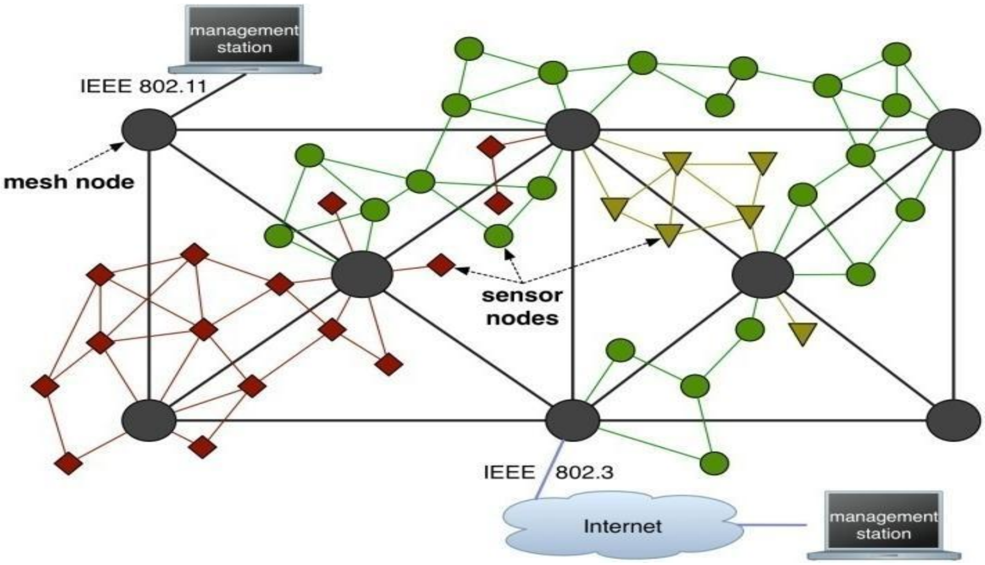

Figure 2: Network Structure of WSN

The nodes of WSN are spatially distributed to monitor physical or environmental parameters. These nodes cooperatively pass their data through the network to a main location. Modern networks are bi-directional – downloading of actuator signal and runtime update of program. Main motivation behind development of WSN was military applications like battlefield surveillance but today they are used in many industrial and consumer applications.

When deployed in large quantities in the field of interest, these sensors ubiquitously acquire data and the system as a whole works like eyes and ears deeply penetrated into the field of interest..The homogeneous nature of the WSN as a whole makes it a perfect candidate for hierarchical network of data gathering elements. The hierarchy of layers also fit well in industrial environment where the process monitoring and control is done using the standard levels of control specified by ISA.

Application Areas of WSN

Wireless sensor networks have been in use for variety of applications now. Few of the applications are briefed here-

In Military applications field WSN have been successfully deployed for information collection, battlefield surveillance enemy tracking, land-mine detection target classification, and tracking of vehicles and armed soldiers. In these applications generally, the input data comes from seismic and acoustic signal sensing.

WSN is used effectively in Disaster relief operations. In place of natural calamities like flood, earthquake etc, WSN can be effectively used to get accurate data for optimizing relief operations. For example, sensor nodes can be dropped from an aircraft over a wildfire so that each node measures surrounding temperature and a “temperature map” can be derived. [15].

WSNs are also used Biodiversity mapping such as tracking of wildlife and exploring their behavior.

WSN have been successfully deployed in intelligent buildings to reduce energy wastage by proper humidity, ventilation, air conditioning (HVAC) control. Such WSNs rely on measurements about room occupancy, temperature, air flow, …, WSNs are also used to monitor mechanical stress in heavy structures like bridges etc to detect need for preventive maintenance.

By embedding sensing/controlling functions inside rotating machines a new era is opened by WSNs in the field of Machine surveillance and preventive maintenance. e.g., tire pressure monitoring has been possible only because of wireless sensing node installed inside a rotating wheel.

There is new branch in agriculture called precision agriculture in which WSN nodes detect soil humidity, green house temperatures, and control precise amount of fertilizer/ pesticides/ irrigation only when and where needed.

In healthcare, long-term surveillance of chronically ill patients or the elderly is done using WSN. Similarly, patients can be mobile if a WSN is deployed to monitor their critical parameters by warable WSN nodes.

Requirement of WSN in Industrial Process

A chemical process plant could be very easily monitored for leaks by hundreds of sensors that automatically form a wireless interconnection network and immediately report the detection of any chemical leaks. Compared to traditional wired systems, the deployment costs of WSN would be very minimal. There is no need to deploy thousands of feet of wire routed through protective conduit. Instead, installers simply have to place quarter-sized device, such as the one pictured in Figure 3, at each sensing point.

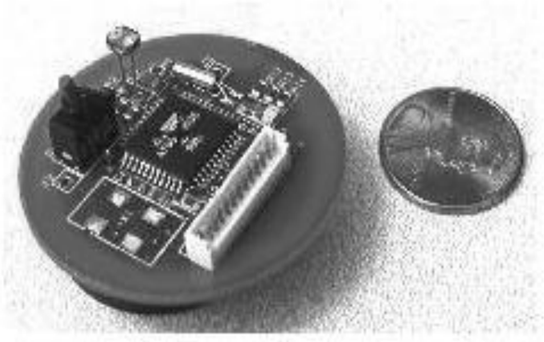

Figure 3: Typical WSN Node (MICA)

The network could be incrementally extended by simply adding more devices – no rework or complex configuration. Normally, the system would be capable of monitoring for anomalies for several years on a single set of batteries. Wireless Sensor Networks not only drastically reduce the installation costs but they also have the ability to dynamically adapt to changing environments. WSNs are flexible enough so that adaptation mechanisms can make it possible to change as per changes in network topologies or can cause the network to shift between drastically different modes of operation. Thus, the same embedded network performing leak monitoring in a chemical factory might be reconfigured into a network designed to localize the source of a leak and track the diffusion of poisonous gases. The WSN could then direct workers to the safest path for emergency evacuation.

WSNs offer distinct advantages when used in Industrial environment for process monitoring and controls –

- Reduced deployment and maintenance costs
- Flexibility in installing/upgrading network
- Suitable for moving and rotating equipment

- Decentralization of automation functions
- Exploitation of micro-electromechanical systems (MEMS): integrated wireless sensors with built-in communication capabilities offer a more robust design than attaching wires to small-sized devices.

ALARM FILTERING

Alarm System

In control room terminology, an alarm is a notification that a fault or an abnormal event has occurred and an operator must take action. Alarms are important for safe and reliable operation of any process plant. An alarm system is a collection of hardware and software that detects, communicates and annunciates alarms to the operator. It consists of everything from detection of an abnormal condition in the plant, generation of alarm, routing the alarm to the control room and giving visible/audible indication of the alarm to the operator. Alarm systems play a key role in informing indications of abnormal process conditions or equipment malfunctions to the operators. Periodic alarm assessment is a crucial step in alarm management lifecycle that provides valuable feedback for fine tuning the alarm system.[14]

Interdependency of Alarms

An abnormal situation in a process plant usually always triggers multiple parameters to go beyond safe threshold values. e.g. overheating of boiler will always give rise to high pressure as well as high temperature. Such interdependence is the root cause of flooding of alarms when an abnormal condition occurs. Alarm flooding is a condition where alarms appear on the control panels at a rate faster than the operator can comprehend or respond to. Alarm flooding overloads and prevents the operator from determining the root cause of the process upset and therefore limits the scope for effective and quick emergency response.

An interdependence of alarms, in its simplest form, can be shown as a directed graph. (Figure 4). In this figure, alarm A1 gives rise to Alarm A2, A4 and A5, whereas alarm A3 raises alarm A2 and A6. So if A6 is generated it can be concluded that alarm A3 is also generated and one of the alarms can be filtered out.

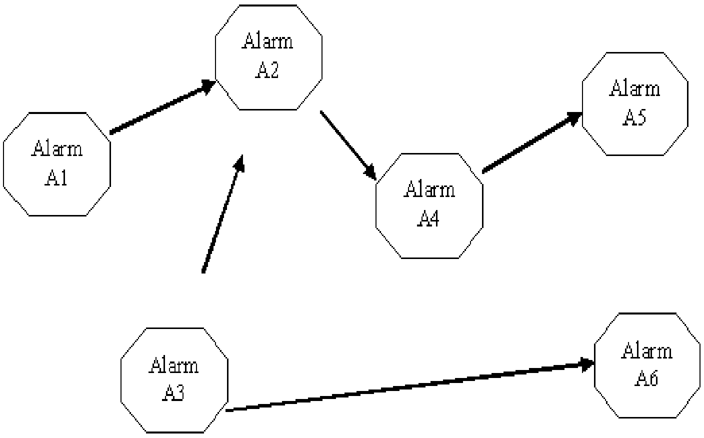

Figure 4: Alarm Interdependence

Necessity of Alarm Filtering

Each alarm must be responded to by the human operator. Therefore, generating large number of alarms is a cognitive burden on the human operator leading to stressful working condition which can be disastrous. An alarm which does not give any new information to the operator should preferably be suppressed so that operator can only focus on the most important alarm and concentrate his energy on removing the root cause of the alarm rather that getting distracted by unnecessary alarm inputs.

The existing researches normally use an expert system for an alarm reduction. Expert systems consist of a knowledge base and an inference engine. The behaviors of an expert system are nondeterministic [6]. Root cause of an alarm is usually a failed system. In a process plant, failure of one device/system may cause another to fail. In this situation, if alarm filtering is not employed, the alarm system will report both the original failure and the other device that failed. Alarm filtering enables the alarm system to report the original failure with more priority than the subsequent failure, allowing the repairman to concentrate on the cause of the issue, and not waste time trying to repair the wrong device.

ALGORITHM FOR ALARM FILTERING

Attributes of an Alarm

An alarm is associated with following attributes - alarm identifier (a plant-wide unique identifier for the alarm, time stamp (this will be the time recorded by the source node of the WSN where first time the alarm got generated., variable name (temperature, pressure, flow etc), tag description, plant name, area, priority of the alarm, alarm set point (the threshold value used for comparing the parameter), value of parameter and so on. Typical alarm looks like this –

A001HI:230520131534:BP: BoilerPressure: Reheating Plant:120:121.5:MICA345261

Here,

A001HI	:	is the unique alarm identifier. HI indicates it’s a high-threshold crossing alarm
230520131534	:	is the time stamp (23rd May, 2013; 15:34 hrs)
BP	:	is the tag which is short form for Boiler Pressure
120	:	Threshold value used for comparing current value of the parameter.
121.5	:	Actual value of the parameter
MICA345261	:	WSN node which recorded this parameter.

All alarm identifiers and associated data are stored in a common database accessible to the sinks and higher level data aggregation layers.

Logic of Alarm Filtering

Building hierarchical relations between alarms The core of alarm filtering logic is the rule base where the alarm interdependence is stored. Alarm dependence can be generically thought as having one or more of the following components-

- **A Logical Combination** (AND, OR, NOT) of a set of alarms or failure conditions – e.g. alarm A001 occurs when A007 AND A003 occur OR A004 occurs in the same time slot.
- **DELAY:** A **time delay** – e.g. 3 units of time slots later than occurrence of alarm B007
- **DUR: Duration** – e.g. Alarm 009 occurs when alarm 006 sustains continuously for certain time period
- **CNT: Count** - e.g. Alarm 011 occurs when alarm 006 sustains continuously for 5 time slots.
- **SEQ : Sequence -** e.g. Alarm A012 occurs if A019 and A020 occur in sequence
- **PROB:** A percentage weightage factor indicating Probability of this condition happening if the predecessor conditions occur to next level of alarm condition

There are many more conditions possible as elaborated in [16] but for this paper, only the simple ones as shown above are considered. The interdependence of alarms can be shown by a tree structure where each higher level is connected with lower level by any of the logical relation mentioned above. The PROB factor is also indicated in circle near the alarm. An example structure of alarm condition A11 is given in Figure 5.

Each alarm is configured to have a list of probable conditions arising from predecessor alarms. Each alarm condition is given a weightage factor (between 0 to 1) which is the probability that this condition will occur provided the predecessor's conditions are satisfied. The weightages are initially heuristically decided based on expertise but the system incorporates a learning function where the weightages will be updated based on history.

There is no limit to number of predecessors. Periodic consistency check is run on the alarm configuration data to remove any anomalies. Each alarm has been associated with a suppression index starting from 0. A higher suppression index indicates that alarm is a possible candidate for filtering out.

Fault Tree The example fault tree in Figure 5 shows following scenario –

- If Alarms A3 and A7 occur one after another (sequentially) then there is a 55% chance that alarm A8 will occur.
- If alarm condition A4 sustains for duration of more than 10 time slots then there is a 35% probability that A9 will occur.
- If A5 occurs and A6 also occurs for more than 2 slots then there is 85% probability that A1 will occur.
- If either A8 or A9 or A1 occur then there is 99% probability that A11 will occur.

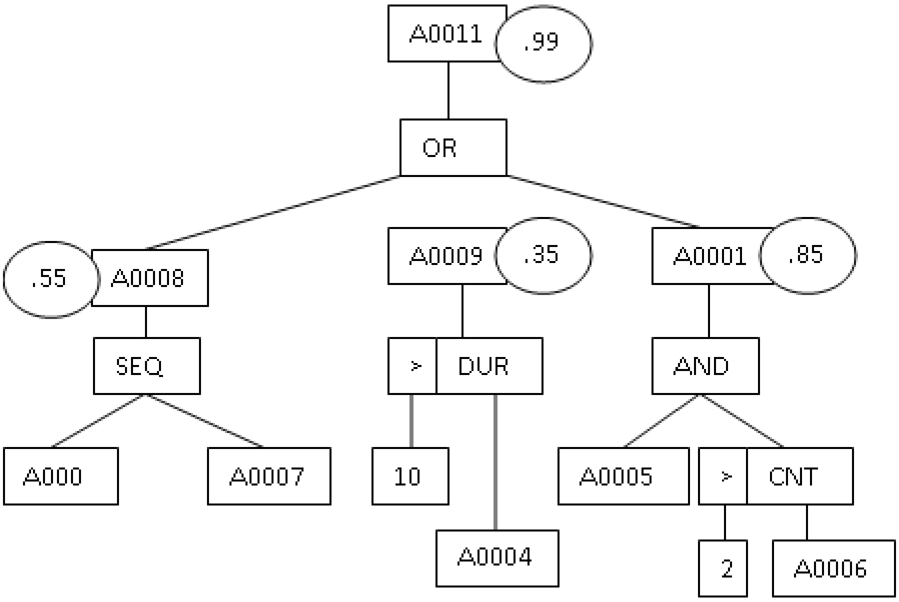

Figure 5: Example of Rule Driven Fault Tree

Alarm prioritization refers to the determination of the relative importance of the alarm with respect to all current alarm conditions, from operator's point of. Alarms determined by the alarm filtering logic to be less important, unnecessary, or otherwise unnecessary are not presented to the operators.

Filtering Algorithm When an alarm occurs, the node checks if any of its predecessor alarm has also occurred. If the sum of the product of the weightages and time differences between the time stamps of all corresponding predecessors which have occurred and the time stamp of the alarm is beyond a threshold value, the suppression index of the alarm is increased by a step. For the purpose of testing we used the threshold of 0.5 and suppression index step size of 1.

Suppose,

Alarm ‘A’ has occurred and its predecessor alarms, which have occurred, are P_1, P_2, P_3 … P_n

Timestamps are denoted by T_A, T_{P1}, T_{P2} etc

Weightages are denoted by W_A, W_{P1}, W_{P2} etc

Then, Sum of weighted differences of time stamps of ‘A’ and its occurred predecessors P1, P2 etc is,

$$\Delta_T = \sum_{i=1}^{i=n} W_{Pi} * (T_A - T_{Pi})$$

If this sum is less than some threshold value, the alarm A is a probable candidate for suppressing. But the suppression of alarm A is not done immediately. We attach a suppression index SA for the Alarm A. It is initialized to zero indicating no alarm is to be suppressed. Every time, Δ < threshold the suppression index for that alarm is increased by a step. When the Suppression index reaches a preset value, the alarm A is suppressed/filtered out at next reporting slot. Such a preset value for our present work is taken as 5. It means that if continuously 5 time slots, it is found that the time stamps of A and its predecessors are very close then on the 6^{th} time slot alarm A will be filtered out.

If the timestamps of an alarm A and its predecessor P1, P2 etc is almost equal (i.e the difference in time stamp is almost 0) then it indicates that these alarms occurred almost simultaneously. This indicates, there is a strong correlation between A and the predecessor alarms.

In such a case, given the occurrence of P1, P2 etc, it can be concluded that alarm A can be suppressed / filtered out. The decision is conditioned by applying the percentage weightage factor which is nothing but heuristic probability of the correlation between A and its predecessors existing. Updating the weightage factors dynamically based on history of alarms is part of the learning which is not implemented in current version.

IMPLEMENTATION ON SIMULATED ENVIRONMENT

RT Linux Environment

The alarm filtering algorithm is tested on simulated environment. Because of Real Time constraints faced in actual environment, it was necessity to use an RTOS as run-time platform. RT-Linux was the obvious choice because of its free availability, huge support in public domain, matured robust and well tested technology and flexibility in real time behavior. RT-Linux was chosen to run with Ubuntu Ver 10.04 (LTS) on a AMD 64-bit 2 GHz machine with 2GB memory.

Each WSN node was created as an independently running RT-Linux thread. The main controller program starts by initializing all the data structures and creating as many RT-linux threads as total no. of individual WSN nodes. The connectivity between the node threads is implemented using named pipes. Delay in communication is simulated by randomly introducing delays in transmission and reception from pipes.

Data Structures and Files

There are three crucial data files which drive the system – “Nodes”, “Alarms” and “Rulebase”

The “nodes” file records list of nodes and associated sensors for each node. It is a plain text file which is read by the main controller program on initialization and dispatches the list of associated sensors of individual nodes to the respective node threads when they are created. The format of the Nodes file is as follows –

Table 1: Nodes Data File (Only Few Typical Entries Mentioned)

Node Identification	Type	Sensors	Primary Connectivity	Secondary Connectivity
SV-0001	Sensor node	Temp-Sen-1 Press-Sen-1 Prox-Sen-5 Temp-Sen-4 Press-Sen-9	SV-1001	SV-0003
SV-0002	Sensor node	Press-Sen-4 Temp-Sen-3 Prox-Sen-1 Prox-Sen-3 Flow-Sens-6	SV-1001	SV-0003
SV-0003	Sensor node	Flow-Sens-5 Press-Sen-5 Temp-Sen-3 Flow-Sens-9	SV-1001	SV-0001
SV-1001	Sink		SV-0001 SV-0002 SV-0003 SV-1002	

The “Alarms” data file stores list of all the alarms with only the basic information about each alarm. It is the main alarm database. The main controller program dispatches the alarms corresponding to individual nodes to the respective threads at the time of creation. The format of the “Alarms” file is as given in Table 2. It stores the unique Alarm ID, the node id which generates the alarm, which sensor generates this alarm, what is the criteria of alarm generation (Less Than / Greater Than etc), what is the set point and what is the hysteresis if any (clearing of the alarm is done only after the value of the parameter goes beyond setpoint+hysteris value). The table shows only few typical entries in the file.

Table 2: Alarms Data File (Only Few Typical Entries Mentioned)

Alarm Identification	Node Generating the Alarm	Sensor Generating the Alarm	Alarm Generation Criteria (LT/GT)	Alarm Set Point	Hysterisis
A-0001	SV-0001	Temp-Sen-4	GT	100	2.0
A-0002	SV-0002	Flow-Sens-6	LT	23	1.0
A-0003	SV-0002	Press-Sen-4	GT	5.5	0.6

The rule base file stores the rules as per logic defined by process SMEs. The structure of the rule base file is as shown in Table 3. Here again only a few typical entries are shown. The rules specified in Figure 5 are tabulated.

Table 3: Rules Database (The Rules Specified in Figure 5 are Tabulated here)

Alarm	Relation with Next Level Predecessor	Next Level Predecessors	Relationship Data	Weightage Factor
A-0008	SEQ	A-0003		55%
		A-0007		
A-0005	CNT	A-0010	5	
A-0009	DUR	-	>10	35%
T-0001	CNT	A-0006	>2	100%
A-0001	AND	A-0005		85%
		T-0001		
A0011	OR	A-0008	-	99%
		A-0009		
		A-0001		

Code Structure

The alarms are generated by the node threads based on random number generator. The nodes communicate the alarms to the nodes of primary connectivity at the specified scanning frequency. The nodes use common system clock. The sink thread runs the alarm filtering algorithm. The main controller program at any point can read suppressed alarms database from the sink thread.

The main controller program maintains record of following parameters.

- Total no. alarms generated (node wise)
- No. of suppressed alarms (at node)
- No. of suppressed alarms (at sink)
- Current values of suppression index (for each Alarm)
- Total no. of rules in the rule base
- Total no. sensor nodes simulated
- Total no. of sensors

RESULTS

Test Data

Testing was carried out with following set of test data –

Table 4: Test Data

Test Set	Parameter	Value
Test Set 1	No. of Nodes Simulated	25
	Average no. of sensors per node	4
	No. of Alarms	135
	No. of Rules	55
	No. of alarms taking part in rules	110
	No. of sinks	5
	Average no. of nodes per sink	5
	Threshold of weighted time stamp difference for triggering suppression index increment	200 msec
	Threshold of suppression index for triggering filter-out	5
Test Set 2	No. of Nodes Simulated	50
	Average no. of sensors per node	4
	No. of Alarms	210
	No. of Rules	98
	No. of alarms taking part in rules	165
	No. of sinks	8
	Average no. of nodes per sink	6
	Threshold of weighted time stamp difference for triggering suppression index increment	200 msec
	Threshold of suppression index for triggering filter-out	5

Observation Scenario

The tests were executed by simulating the WSN as per test data. The data files were created as text files and the controller code reads those file as part of initialization process and populates the internal data structures. The testing is done by not only the two data sets given above but also continuously varying the individual parameters and testing. e.g. No. of nodes varying from 25 to 50 in steps of 8; and no. of rules varying from 55 to 98 in steps of 10. The controller program calculates the no. of suppressed alarms and records them. Testing was also carried out by completely bypassing rule base (i.e.

no alarm suppression), and also by removing the probability factors (making all weightage factors 100%).

At the end, before exiting, the program writes the findings in a text file along with time stamp of test execution. It also prints out the results on screen.

RESULTS

Following results were obtained:

Table 5: Results Obtained (Only Representative Data Reproduced here)

No. of Nodes	No. of Rules	Suppression Index Threshold	Threshold of Time Stamp Difference	% of Alarms Suppressed
25	55	5	200 msec	5%
25	60	5	200 msec	7%
25	70	5	200 msec	10%
25	80	5	200 msec	12%
25	98	5	200 msec	20%
40	55	5	200 msec	10%
40	60	5	200 msec	15%
40	70	5	200 msec	17%
40	80	5	200 msec	20%
40	98	5	200 msec	23%
50	55	4	200 msec	19%
50	60	4	200 msec	25%
50	70	4	200 msec	29%
50	80	4	200 msec	34%
50	98	4	200 msec	40%

CONCLUSIONS AND FURTHER WORK

The conclusions drawn from the results are summarized below.

- Number of alarms suppressed expressed as percentage of total alarms can be approximately taken as performance indicator of suppression algorithm.
- The algorithm starts giving better result when - no. of nodes increase, no. alarms increase and no. rules increase.
- Quality of rules decides how effective the suppression algorithm is working.
- Reduction in suppression index threshold for triggering filtering gives higher percentage of alarms reduced.
- The threshold of timestamp difference does not have much impact on suppression logic performance. But this could be because of simulated environment. In real life scenario it my give different results.

The Alarm system is one of the most crucial one in Industrial Process Control. In fact, alarms are the main connection between the operator and the automation. When the process is operating outside its normal boundaries, such operation requires tools that allow the operator to effectively call on his/her expertise to look through the labyrinth of alarms and concentrate on those which are most important and high priority. We presented in this paper, a method of reducing superfluous and duplicated alarms so that operator cognitive burden is reduced and operator can concentrate on crucial issue first.

Further work in this area may involve building a GUI tool to input rules. This can be easily done using HLPN representation [16]. Expert system has always been a hot topic reaseach and sophisticated Alarm filtering comes under the realms of expert system. Developing rules interconnecting Alarms required deep knowledge of the process. More data are necessary to assess fully the performance of the model proposed. The developments in this field can be used in applications such as avionics fault diagnosis, automotive, manufacturing and transportation systems.

REFERENCES

1. Chao Wang, Huadong Ma, Yuan He, Shuguagn Xiong, “Adaptive Approximate Data Collection for Wireless Sensor Networks”, IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS, VOL. 23, NO. 6, **JUNE 2012**

2. Chao Wang, Huadong Ma, Yuan He, Shuguagn Xiong, “A Dynamic En-route Filtering Scheme for Data Reporting in Wireless Sensor Networks”, IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS, VOL. 23, NO. 6, **JUNE 2012**

3. Chi-Tsun Cheng,, Chi K. Tse, Francis C. M. Lau, “An Energy-Aware Scheduling Scheme for Wireless Sensor Networks”, IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY, VOL. 59, NO. 7, **SEPT 2010**

4. Tossaporn Srisooksai, Kamol Keamarungsi, Poon lap Lamsrichan, Kiyomichi Araki, “Practical data compression in wireless sensor networks: A survey”, **Mar 2011**

5. Marko Paavola and Kauko Leiviska (2010). Wireless Sensor Networks in Industrial Automation, Factory Automation, Javier Silvestre-Blanes (Ed.), ISBN: 978-953-307-024-7.

6. Gwi-sook Jang, Sang-moon Suh, Sa-kil Kim, Yong-suk Suh, Je-yun Park, “A proactive alarm reduction method and its human factors validation test for a main control room for SMART”, **Oct 2012**

7. E.H.Bristol, “Improved Process Control Alarm Operation”, ISA Transactions, June 2001

8. Zineb Simeu-Abazi, Arnaud Lefebvre, Jean-Pierre Derain, “A methodology of alarm filtering using dynamic fault tree”, Journal of Reliability Engineering and System Safety, Elsevier Publications, **Nov 2010**

9. Daojing He, Sammy Chan, “SDRP: A Secure and Distributed Reprogramming Protocol for Wireless Sensor Networks”, IEEE TRANSACTIONS ON INDUSTRIAL ELECTRONICS, **Nov 2012**

10. Elhadi M. Shakshukia,∗, Haroon Malikb, Tarek R. Sheltamic, “A comparative study on simulation vs. real time deployment in wireless sensor Networks”, The Journal of Systems and Software, Elsevier Publications, **Aug 2010**

11. Miao Liu, Duo Liu, Yi Wang, Meng Wang, and Zili Shao, “On Improving Real-Time Interrupt Latencies of Hybrid Operating Systems with Two-Level Hardware Interrupts”, IEEE TRANSACTIONS ON COMPUTERS, VOL. 60, NO. 7, **JULY 2011**

12. Jianzhong Li, Member, IEEE, Lei Yu, Member, IEEE, Hong Gao, and Shuguang Xiong, “Grouping-Enhanced Resilient Probabilistic En-Route Filtering of Injected False Data in WSNs”, IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS, VOL. 23, NO. 5, **MAY 2012**

13. Dan Wu, Man Hon Wong, “Fast and Simultaneous Data Aggregation Over Multiple Regions in Wireless Sensor Networks”, IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS—PART C: APPLICATIONS AND REVIEWS, **MAY 2011**

14. Sandeep R. Kondaveeti, Iman Izadi, Sirish L. Shah, Tim Black, Tongwen Chen, “Graphical tools for routine assessment of industrial alarm systems”, The journal of Computers and Chemical Engineering, Elsevier Publications, **JULY 2012**

15. Th. Arampatzis, J. Lygeros, S. Manesis. “A Survey of Applications of Wireless Sensors and Wireless Sensor Networks”, 13th Mediterranean Conference on Control and Automation, **JUNE 2005**

16. Zineb Simeu-Abazi, ArnaudLefebvre, Jean-PierreDerain, “A methodology of alarm filtering using dynamic fault tree”,Journal of Reliability Engineering and System Safety, **NOV 2010**.